

# Compiling on macOS

## OPENCN - COMPILING FOR MACOS

Revised: 11:00, 3 October 2019

### XCODE (Version 11)

Download Xcode\_11.xip from the Apple Developer website (<https://developer.apple.com/download/more/>) - requires registration (free) at developer.apple.com website. Unzip and install Xcode\_11

Download Command\_Line\_Tools\_for\_Xcode\_11.dmg from the Apple Developer website (<https://developer.apple.com/download/more/>). Unzip and install Command\_Line\_Tools\_for\_Xcode\_11

The OpenCPN official build supports OS () X 10.9 and newer. Regardless of Xcode version, older OS () X SDK is not needed to build with support for older versions of macOS, but support for older macOS versions has to be explicitly turned on during the build of OpenCPN, wxWidgets and other libraries as described below.

### DEVELOPMENT TOOLS

There are required development tools that can be installed via Homebrew (<https://brew.sh>) (or similarly from MacPorts, but it is unsupported and untested).

Install Home-brew:

```
$ /usr/bin/ruby -e "$(curl -fsSL
https://raw.githubusercontent.com/Homebrew/install/master/install)"
```

Install cmake and gettext via Homebrew:

```
$ brew install cmake
$ brew install gettext
```

Add the gettext tools to your PATH environment variable using:

```
$ echo 'export PATH="/usr/local/opt/gettext/bin:$PATH"'>> ~/.bash_profile
```

Install Tinyxml:

```
$ brew install tinyxml
```

And either restart Terminal.app or reload your environment using

```
. ~/.bash_profile
```

## REQUIRED EXTERNAL LIBRARIES

If you are building OpenCPN with SVG support (default), Cairo library is needed, install it using:

```
$ brew install cairo
```

Additionally, to support all the features of the wxSVG component, libexif should be installed using:

```
$ brew install libexif
```

If you are building OpenCPN with extended archive support (default, required to download and unpack up to date GSHHG basemaps and pilot charts using the chart downloader plugin), libarchive library is needed, install it using:

```
$ brew install xz libarchive
```

[Read following note - install only if necessary] - To produce binaries fully compatible with older macOS versions on newer macOS, libarchive must instead of from Homebrew, be built as follows:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9
$ wget https://libarchive.org/downloads/libarchive-3.3.3.tar.gz
$ tar xzf libarchive-3.3.3.tar.gz
$ cd libarchive-3.3.3
$ ./configure --without-lzo2 --without-nettle --without-xml2 --without-openssl --with-expat make
$ make install
$ cd ..
```

## wxWIDGETS

OpenCPN is built upon wxWidgets, a package providing a cross-platform UI SDK. But to get results fully compatible with the official OpenCPN builds, which retain compatibility with macOS Mavericks (10.9) and newer, wxWidgets must be version 3.1.2 built manually as follows: Download wxWidgets-3.1.2.tar.bz2 from: <https://www.wxwidgets.org/news/2018/12/wxwidgets-3.1.2-released/> Unzip and move wxWidgets-3.1.2 folder to Desktop

Install wxWidgets 3.1.2:

```
$ mkdir build-release && cd build-release
$ ../configure --with-cxx=11 --with-macosx-version-min=10.9 --enable-unicode
--with-osx-cocoa --enable-aui --disable-debug --with-opengl
$ make -j2
$ sudo make install
```

## PACKAGES TOOL

To create .pkg files for plugins, download the “Packages” tool from <http://s.sudre.free.fr/Software/Packages/about.html> (<http://s.sudre.free.fr/Software/Packages/about.html>). Install Packages.dmg by double-clicking the file and following the instructions - no need to open Packages.app as the installer configures the necessary command line tools for Packages.

Check Configuration for problems or errors:

```
$ brew doctor
```

If everything is installed OK, you should get the message “Your system is ready to brew”

## CREATING AN INSTALLABLE PACKAGE AND INSTALLING OPENCNP

Officially OpenCPN is built from the command line, but it's also possible to use Apple's Xcode IDE for development. Regardless of which method we choose, the first steps are the same: Get the OpenCPN source:

```
$ git clone https://github.com/OpenCPN/OpenCPN.git
```

Create the build directory, where our local builds will take place, so that we don't work directly in the source tree:

```
$ mkdir OpenCPN/build && cd OpenCPN/build
```

**Build Method 1** - From the command line Prepare our build environment:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9
$ cmake ..
```

Build OpenCPN:

```
$ make
```

To build a debug version use:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9
```

```
$ cmake -DCMAKE_BUILD_TYPE=Debug ..  
$ make
```

## **Build Method 2** - Using Xcode Create the Xcode project:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9  
$ cmake -G Xcode ..
```

Open the `OpenCPN.xcodeproj` file in Xcode, and use the “Build”, “Run”, “Debug”, etc features as normal. To use the “Run” action you need to build the “OpenCPN” target rather than the default “ALL\_BUILD” target.

Build and install OpenCPN:

```
$ make install
```

## **WARNING** - Do The Following:

The default install location is (/usr/local/bin). Everything from /usr/local/bin get's packaged into your DMG which is not desirable. To avoid this, change the install location with 'cmake' as follows:

```
$ cmake -DCMAKE_INSTALL_PREFIX=/Users/dsr/tmp ..
```

Some developers have reported that the install step copies a redundant set of the wxWidgets dynamic library into the install directory, causing OpenCPN to fail. This is intended, but gets annoying for local bundles not intended to be distributed. A kludgey fix:

```
$ sudo rm /usr/local/bin/OpenCPN.app/Contents/MacOS/libwx*.dylib
```

Build the installable DMG:

```
$ make create-dmg
```

Depending on your local system, during both steps above you may observe insufficient permissions on some files. Either fix the permissions or use sudo to run make install/create-dmg

To install the application, double-click on the DMG in Finder and drag OpenCPN.app to the Applications directory.

## **BUILDING PLUGINS (example)**

Building Watchdog\_pi from dev branch:

Get source code:

```
$ git clone git://github.com/seandepagnier/watchdog_pi
```

Build from command line:

```
$ mkdir ~/watchdog_pi/build && cd ~/watchdog_pi/build  
$ export MACOSX_DEPLOYMENT_TARGET=10.09  
$ cmake ..  
$ make  
$ make create-pkg
```

Double-click on the package in ~/watchdog\_pi/build/Watchdog-Plugin-ov50\_2.4.pkg This installs into /Applications/OpenCPN.app

## EARLIER INSTRUCTIONS

## Compiling v5.0

These instructions are valid for the current codebase, *if you need to build OpenCPN 4.8.x, please refer to [historic version of this page](#)* .

### Xcode

- Install Xcode from the Mac App Store (free registration at [developer.apple.com](https://developer.apple.com) required)
- Install Command Line Tools for Xcode (available from [developer.apple.com](https://developer.apple.com))

The OpenCPN official build supports OS X 10.9 and newer. Regardless of Xcode version, older OS X SDK is not needed to build with support for older versions of macOS, but support for older macOS versions has to be explicitly turned on during the build of OpenCPN, wxWidgets and other libraries as described below.

### Development Tools

There are required development tools that can be installed via [Homebrew](#) (or similarly from MacPorts, but it is unsupported and untested).

- Install cmake and gettext via Homebrew

```
$ brew install cmake  
$ brew install gettext
```

Add the gettext tools to your PATH environment variable using

```
$ echo 'export PATH="/usr/local/opt/gettext/bin:$PATH"'>> ~/.bash_profile
```

and either restart Terminal.app or reload your environment using

```
. ~/.bash_profile
```

- Install the “Packages” tool for creating .pkg files for plugins from <http://s.sudre.free.fr/Software/Packages/about.html>.

## Required external libraries

If you are building OpenCPN with SVG support (default), Cairo library is needed, install it using

```
$ brew install cairo
```

additionally, to support all the features of the wxSVG component, libexif should be installed using

```
$ brew install libexif
```

If you are building OpenCPN with extended archive support (default, required to download and unpack up to date GSHHG basemaps and pilot charts using the chart downloader plugin), libarchive library is needed, install it using

```
$ brew install xz libarchive
```

To produce binaries fully compatible with older macOS versions on newer macOS, libarchive must instead of from Homebrew built as follows

```
export MACOSX_DEPLOYMENT_TARGET=10.9
wget https://libarchive.org/downloads/libarchive-3.3.3.tar.gz
tar xzf libarchive-3.3.3.tar.gz
cd libarchive-3.3.3
./configure --without-lzo2 --without-nettle --without-xml2 --without-openssl -
-with-expat
make
make install
cd ..
```

## wxWidgets

OpenCPN is built upon wxWidgets, a package providing a cross-platform UI SDK.

*But to get results fully compatible with the official OpenCPN builds, which retain compatibility with*

macOS Mavericks (10.9) and newer, *wxWidgets must be version 3.1.2 built manually* as follows:

```
$ mkdir build-release
$ cd build-release
$ ../configure --with-cxx=11 --with-macosx-version-min=10.9 --enable-unicode -
-with-osx-cocoa --enable-aui --disable-debug --with-opengl --without-subdirs
$ make -j2
$ sudo make install
```

## Building OpenCPN

Officially OpenCPN is built from the command line but it's also possible to use Apple's Xcode IDE for development. Regardless of which method we choose, the first steps are the same:

- Get the OpenCPN source:

```
$ git clone https://github.com/OpenCPN/OpenCPN.git
```

Create the build directory, where our local builds will take place, so that we don't work directly in the source tree:

```
$ mkdir OpenCPN/build && cd OpenCPN/build
```

### Build Method 1 - From the command line

Prepare our build environment:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9
$ cmake ..
```

Build OpenCPN:

```
$ make
```

To build a debug version use:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9
$ cmake -DCMAKE_BUILD_TYPE=Debug ..
$ make
```

### Build Method 2 - Using Xcode

Create the Xcode project:

```
$ export MACOSX_DEPLOYMENT_TARGET=10.9
$ cmake -G Xcode ..
```

Open the `OpenCPN.xcodeproj` file in Xcode, and use the “Build”, “Run”, “Debug”, etc features as normal. To use the “Run” action you need to build the “OpenCPN” target rather than the default “ALL\_BUILD” target.

## Installing OpenCPN and Creating an Installable Package

- Build and install OpenCPN:

```
$ make install
```

The default install location (/usr/local/bin) can be changed with cmake (**And should be** in case you want to create the DMG image, if you don't change it, everything from /usr/local/bin get's packaged into your DMG. You have been warned.):

```
$ cmake -DCMAKE_INSTALL_PREFIX=/Users/dsr/tmp ..
```

Some developers have reported that the install step copies a redundant set of the wxWidgets dynamic library into the install directory, causing OpenCPN to fail. This is of course intended, but gets annoying for local bundles not intended to be distributed. A kludgy fix:

```
$ sudo rm /usr/local/bin/OpenCPN.app/Contents/MacOS/libwx*.dylib
```

- Build the installable DMG:

```
$ make create-dmg
```

Depending on your local system, during both steps above you may observe insufficient permissions on some files. Either fix the permissions or use sudo to run make install/create-dmg

To install the application, double-click on the DMG in Finder and drag OpenCPN.app to the Applications directory.

From:  
<https://opencpn.org/wiki/dokuwiki/> - **OpenCPN Manuals**

Permanent link:  
[https://opencpn.org/wiki/dokuwiki/doku.php?id=opencpn:developer\\_manual:developer\\_guide:compiling\\_mac\\_osx](https://opencpn.org/wiki/dokuwiki/doku.php?id=opencpn:developer_manual:developer_guide:compiling_mac_osx)

Last update: **2021/01/29 13:59**

